International Journal of Approximate Reasoning 49 (2008) 689–700

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

journal homepage: www.elsevier.com/locate/ijar

Rough sets approach to symbolic value partition

Fan Min *,1, Qihe Liu 2, Chunlan Fang 3

School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China

ARTICLE INFO

Article history:
Received 9 August 2007
Received in revised form 22 April 2008
Accepted 17 July 2008
Available online 25 July 2008

Keywords:
Rough sets
Attribute reduction
Continuous attribute discretization
Symbolic value partition
Scaling
Reduct
Partition reduct

ABSTRACT

In data mining, searching for simple representations of knowledge is a very important issue. Attribute reduction, continuous attribute discretization and symbolic value partition are three preprocessing techniques which are used in this regard. This paper investigates the symbolic value partition technique, which divides each attribute domain of a data table into a family for disjoint subsets, and constructs a new data table with fewer attributes and smaller attribute domains. Specifically, we investigates the optimal symbolic value partition (OSVP) problem of supervised data, where the optimal metric is defined by the cardinality sum of new attribute domains. We propose the concept of partition reducts for this problem. An optimal partition reduct is the solution to the OSVP-problem. We develop a greedy algorithm to search for a suboptimal partition reduct, and analyze major properties of the proposed algorithm. Empirical studies on various datasets from the UCI library show that our algorithm effectively reduces the size of attribute domains. Furthermore, it assists in computing smaller rule sets with better coverage compared with the attribute reduction approach.

1. Introduction

Many problems in machine learning, pattern recognition, and signal process involve high dimensional descriptions of data. In some applications, such as text process and Web content classification, data sets are often represented in the form of information or decision tables with, not only a huge number of attributes (in the order of tens of thousands) [1], but also large cardinalities of attribute domains (in the order of hundreds). Utilizing most generalization techniques, such as rule induction [2], or decision tree construction [3], on such data is rather difficult and often infeasible.

It is, therefore, desirable to develop preprocessing techniques to reduce both the number of attributes and the cardinalities of attribute domains. Another important motivation for these techniques is that lowering the degree of precision in the data makes the data pattern more visible [4].

Attribute reduction, continuous attribute discretization and symbolic value partition are three such techniques. With respect to Rough sets, attribute reduction methods are based on the observation that attributes are not independent and consequently, some of them are superfluous [5]. Researchers have proposed a number definitions for reducts (see, e.g., [5–9]) to provide different levels of information reservation, together with many heuristic algorithms (see, e.g., [10–12]), that have been successfully applied in numerous applications. However, this technique can only reduce the number of attributes,

* Corresponding author. Tel.: +86 13540685200.
E-mail addresses: minfan@uestc.edu.cn (F. Min), qiheliu@uestc.edu.cn (Q. Liu), fangcl@uestc.edu.cn (C. Fang).
URL: http://www.sccua.cn/show.aspx?ID=75 (F. Min).

1 Fan Min is supported by an information distribution project under Grant No. 9140A06060106DZ223 and the Youth Foundation of UESTC.
2 Qihe Liu is supported by the National Science Foundation of China under Grant No. 60702071.
3 Chunlan Fang is supported by the China 863 project under Grant No. 2007AA01Z322.

0888-613X/$ - see front matter
doi:10.1016/j.ijar.2008.07.002

and is thus unsuitable for continuous attributes or attributes with large domains [13]. Although continuous attribute discretization has been investigated by a wider range of research groups (see, e.g., [14–16]) than that of attribute reduction, there are still more studies to be done.

Symbolic value partition is more general, and thus more complex than attribute reduction and continuous attribute discretization. In fact, both the attribute reduction problem and the continuous attribute discretization problem are special cases of the partition problem [15]. Moreover, given an information system with N attributes, each of which has M attribute values, the search spaces of the attribute reduction problem, the continuous attribute discretization problem and the symbolic value partition problem (or simply the partition problem for the sake of brevity) are 2^N, $2^{N(M-1)}$, and $(M!2^{M-1})^N$, respectively. Thus, only heuristic algorithms are applicable.

In the decision tree research field, symbolic value partition is also referred to as attribute value grouping [17–19]. It was provided as an option in C4.5 [17]. Berkman [18] reported that this approach has been shown to have no strong effect on classification accuracy. In addition, this kind of approaches [17,15,19] helps obtain smaller decision trees, that are easier to understand.

From a global viewpoint, we are interested in *optimal consistent partition schemes*, in which the optimal metric is defined by the cardinality sum of new attribute domains, and consistency means preserving the discernibility relation between objects from different decision classes [20]. H.S. Nguyen and S.H. Nguyen [15,21] introduced the concept of optimal symbolic value partition (OSVP) problem and proposed two approaches to address the problem, namely a decision tree approach and a Rough sets approach. The first approach, known as the MDG-method, addresses the binary optimal partition (BOP) problem, by partitioning each attribute value set into two disjoint subsets in a top-down manner, until some terminating conditions hold. The second approach is interesting in that it converts the initial problem into a graph coloring problem. It does, however, have a few drawbacks, as analyzed in Section 5. In fact, the partition problem has not been investigated thoroughly, and deserves more in-depth research.

In this paper, we propose the concept of partition reducts since finding *optimal* consistent partition schemes is computationally infeasible. The concepts of an optimal consistent partition scheme and an optimal partition reduct are equivalent. We develop an algorithm, all possible outputs of which form the set of all partition reducts. The main idea is to convert the partition problem into a series of attribute reduction problems. We point out that an algorithm, all possible outputs of which choosing optimal or suboptimal reducts in each turn, a suboptimal partition reduct can gradually be obtained. Moreover, since the generalization ability of symbolic value partition is stronger than that of attribute reduction, our algorithm can assist in obtaining smaller decision tables and smaller decision rule sets with better predication ability compared with some existing reduction algorithms.

The rest of the paper is organized as follows: Section 2 describes some basic concepts that are used throughout the paper. Section 3 proposes an algorithm called the reduction-based symbolic value partition (RBSVP) algorithm for the partition problem. Section 4 validates our analysis through experiments. The approach used by H.S. Nguyen and S.H. Nguyen [15,21] is very interesting, yet totally different to our approach. As such we analyze it and give a counterexample in Section 5. Finally, Section 6 presents our conclusions and points out further works.

2. Preliminaries

In this section, describe related relative concepts including decision tables [22], partition schemes [15,21], M-reducts [23] and scaling [24]. Some well known concepts such as the indiscernibility relation, positive region and relative reducts [5,20] are used without further explanation.

2.1. Decision tables

Data are often presented as a table, columns of which are labelled by *attributes*, rows by *objects* of interest and entries in the table are *attribute values*. This paper is only concerned with decision tables having one decision attribute. Formally, a *decision table* is a triple $S = (U, C, \{d\})$, where $d \notin C$ is the decision attribute and elements of C are called *conditional attributes* or *conditions* in brief. Table 1 gives a decision table, where $U = \{x_1, x_2, \ldots, x_9\}$, $C = \{\text{occupation}, \text{ temperature}, \text{ cough}\}$ and $d = \text{SARS}$. This table is referred to as an example throughout the paper.

2.2. M-relative reduct

The number of relative reducts of a decision table may be large [23,25], and many minimal reducts of a decision table may exist. Bazan presented the concept of dynamic reducts [6] to obtain *stable* reducts.

To obtain more preferred reducts, we previously proposed the concept of an M-relative reduct [23] to that an attribute set specified by the user or the algorithm is always included. Since this concept will be used in Section 3, we define it as follows:

Definition 1. Given a decision table $S = (U, C, \{d\})$ and a set of specified attributes $M \subseteq C$, any $B \subseteq C$ is called an M-relative reduct of S iff:

(1) $M \subseteq B$;
(2) $\mathrm{POS}_B(\{d\}) = \mathrm{POS}_C(\{d\})$;
(3) $\forall a \in (B - M),\ \mathrm{POS}_{B-\{a\}}(\{d\}) \subset \mathrm{POS}_C(\{d\})$.

The set of all M-relative reducts of S is denoted by $\mathrm{Red}(S, M)$.

2.3. Partitions

To facilitate our discussion, we use more general definitions than those used in [21]. Note that most of our definitions are essentially equivalent to existing ones.

Let $S = (U, C, \{d\})$ be a decision table, where $C = \{a_i : U \to V_{a_i}\}$ for $i \in \{1, \ldots, |C|\}$. Any function $P_i = P_{a_i} : V_{a_i} \to W_{a_i} \cup V_{a_i}$, where $W_{a_i} \cap V_{a_i} = \emptyset$, $P_i(v) \in W_{a_i}$ or $P_i(v) = v$ is called *a partition of* V_{a_i}. The function P_i defines a new *partition attribute* $a_i^{P_i} = P_i \circ a_i$, i.e., $a_i^{P_i}(u) = P_i(a_i(u))$ for any object $u \in U$. The range of $a_i^{P_i}$ is $V_{a_i}^{P_i} = \bigcup_{u \in U}\{a_i^{P_i}(u)\}$. Often P_i is also expressed as a set of value pairs, i.e., $P_i(v_1) = v_2 \iff (v_1, v_2) \in P_i$.

For example, $P_1 = \{(\mathtt{student}, 1), (\mathtt{doctor}, 2), (\mathtt{nurse}, 2), (\mathtt{teacher}, \mathtt{teacher}), (\mathtt{lawyer}, 1)\}$ is a partition of V_{a_1}, where $W_{a_1} = \{1, 2\}$ and a_1 is an attribute of S given in Table 1. P_1 divides V_{a_1} into three subsets: {student, lawyer}, {doctor, nurse} and {teacher}. $V_{a_1}^{P_1} = \{1, 2, \mathtt{teacher}\}$.

Any array of partitions $P = [P_1, \ldots, P_{|C|}]$ is called a partition scheme of S. P defines from S a new decision table $S^P = (U, C^P, \{d\})$, where $C^P = \{a_1^{P_1}, \ldots, a_{|C|}^{P_{|C|}}\}$. The rank of S is the value $\sum_{i=1}^{|C|}|V_{a_i}|$. The rank of P_{a_i} is the value $\mathrm{rank}(P_{a_i}) = |V_{a_i}^{P_i}|$.

Similar to the definition of a reduct in [5], we propose the following two concepts. A partition scheme P is *consistent* iff $\mathrm{POS}_{C^P}(\{d\}) = \mathrm{POS}_C(\{d\})$. A decision table S is *unpartitionable* iff no partition scheme P exists such that P is consistent and $\mathrm{rank}(S^P) < \mathrm{rank}(S)$.

Definition 2. P is called a *partition reduct* of S iff P is consistent and S^P is unpartitionable.

The set of all partition reducts of S is denoted by $\mathrm{PR}(S)$. A partition reduct P is *optimal* iff $\mathrm{rank}(S^P)$ is minimal. We consider the following problem, which was proven NP-hard [21]:

Problem 3. *Optimal symbolic value partition (OSVP)*
Input: A decision table $S = (U, C, \{d\})$ where all attributes are symbolic.
Output: An optimal partition reduct P of S.

2.4. Scaling

In some theories, especially formal context analysis [24], there is a need to transform a many-valued attribute into a number of binary-valued attributes. This process is called scaling [24]. Here, we require that the decision attribute is not changed in the scaling process.

Definition 4. Given a decision table $S = (U, C, \{d\})$, the set of scaled attributes of $Q \subseteq C$ is

$$Q_B = \{(a, v) | a \in Q, v \in V_a\}, \tag{1}$$

where $(a, v) : U \to \{0, 1\}$ and

$$(a, v)(u) = \begin{cases} 1 & \text{if } a(u) = v; \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Table 2 gives the scaled decision table $S_B = (U, C_B, \{d\})$ of S given in Table 1, where $(O, \mathrm{s}), (O, \mathrm{d}), \ldots, (C, \mathrm{n})$ stand for (occupation, student), (occupation, doctor),…,(cough, no), respectively.

Table 1
An example decision table S

U	Occupation	Temperature	Cough	SARS
x_1	Student	Low	Yes	Suspicious
x_2	Doctor	High	No	Yes
x_3	Nurse	High	Yes	Yes
x_4	Nurse	Normal	Yes	Yes
x_5	Teacher	Normal	No	Suspicious
x_6	Teacher	Normal	Yes	Suspicious
x_7	Lawyer	Normal	Yes	No
x_8	Student	Normal	No	No
x_9	Student	High	No	No

Table 2
S_B, the scaled decision table of S

	$(O,\mathtt{s})$	$(O,\mathtt{d})$	$(O,\mathtt{n})$	$(O,\mathtt{t})$	$(O,\mathtt{l})$	$(T,\mathtt{l})$	$(T,\mathtt{h})$	$(T,\mathtt{n})$	$(C,\mathtt{y})$	$(C,\mathtt{n})$	d
x_1	1	0	0	0	0	1	0	0	0	1	Suspicious
x_2	0	1	0	0	0	0	1	0	0	1	Yes
x_3	0	0	1	0	0	0	1	0	1	0	Yes
x_4	0	0	1	0	0	0	0	1	1	0	Yes
x_5	0	0	0	1	0	0	0	1	0	1	Suspicious
x_6	0	0	0	1	0	0	0	1	1	0	Suspicious
x_7	0	0	0	0	1	0	0	1	1	0	No
x_8	1	0	0	0	0	0	0	1	0	1	No
x_9	1	0	0	0	0	0	1	0	0	1	No

The scaling process has no influence on the indiscernibility relations or positive regions of attribute sets. This is given by the following lemma:

Lemma 5. *Given a decision table $S = (U, C, \{d\})$ for any $i \in \{1, \ldots, |C|\}$,*

$$\mathrm{Ind}(\{a_i\}) = \mathrm{Ind}(\{a_i\}_B), \tag{3}$$
$$\mathrm{Ind}(C) = \mathrm{Ind}(C_B), \tag{4}$$
$$\mathrm{POS}_C(\{d\}) = \mathrm{POS}_{C_B}(\{d\}). \tag{5}$$

Proof. We only prove Eq. (4). According to the construction of C_B $\forall a_i \in C$ and $u_j, u_k \in U$,

$$a_i(u_j) = a_i(u_k) \Longleftrightarrow \forall (a_i, v) \in C_B, (a_i, v)(u_j) = (a_i, v)(u_k).$$

Hence, $\mathrm{Ind}(a_i) = \bigcap_{(a_i,v)\in C_B} \mathrm{Ind}((a_i, v))$, $\mathrm{Ind}(C) = \bigcap_{1\leqslant i\leqslant |C|} \mathrm{Ind}(a_i) = \bigcap_{1\leqslant i\leqslant |C|} (\bigcap_{(a_i,v)\in C_B} \mathrm{Ind}((a_i, v))) = \bigcap_{(a,v)\in C_B} \mathrm{Ind}((a, v)) = \mathrm{Ind}(C_B)$. □

3. The reduction-based symbolic value partition algorithm

In this section, we first explain the main idea of the algorithm using an example. Then we propose the OSGP-problem as the basis of our algorithm and list the algorithm. Finally we analyze various properties of the algorithm.

3.1. An example

To make our algorithm easier to comprehend, we analyze an example prior to any theoretic analysis.

One can apply attribute reduction to the scaled decision table. For example, $R^1 = \{(O, \mathtt{d}), (O, \mathtt{n}), (O, \mathtt{t}), (T, \mathtt{l})\}$ is a minimal reduct of S_B given in Table 2 and a reduced decision table $(U, R^1, \{d\})$ can be obtained. However, it is more interesting to convert $(U, R^1, \{d\})$ back to a "normal" decision table as given in Table 3, in which duplicate objects have been are removed.

Because $(O, \mathtt{s}) \notin R^1$ and $(O, \mathtt{l}) \notin R^1$, we do not distinguish `student` from `lawyer`. From a semantic point of view, `student` and `lawyer` can be replaced by others. However, for the extended purpose, 1 is used instead.

In fact, a partition scheme $P^1 = [\{(\mathtt{s}, 1), (\mathtt{d}, \mathtt{d}), (\mathtt{n}, \mathtt{n}), (\mathtt{t}, \mathtt{t}), (\mathtt{l}, 1)\}, \{(\mathtt{l}, \mathtt{l}), (\mathtt{h}, 1), (\mathtt{n}, 1), \{(\mathtt{y}, 1), (\mathtt{n}, 1)\}]$ could be constructed such that the decision table given in Table 3 is just S^{P^1}.

Another key idea is that this process of scaling, reduction and converting back (in which new values such as 2,3,... are used instead of 1) can be repeated until the cardinality of any attribute cannot be reduced further.

$S_B^{P^1}$ is obtained as given in Table 4. $R^2 = \{(O^{P^1}, 1), (O^{P^1}, t), (T, 1)\}$ is a reduct of $S_B^{P^1}$, and S^{P^2} is obtained as given in Table 5, where $P^2 = [\{(\mathtt{s}, 1), (\mathtt{d}, 2), (\mathtt{n}, 2), (\mathtt{t}, \mathtt{t}), (\mathtt{l}, 1)\}, \{(\mathtt{l}, 2), (\mathtt{h}, 1), (\mathtt{n}, 1), \{(\mathtt{y}, 1), (\mathtt{n}, 1)\}]$. $S_B^{P^2}$ is given in Table 6. $R^3 = \{(O^{P^2}, 1), (O^{P^2}, 2), (T^{P^2}, 1)\}$ is a reduct of $S_B^{P^2}$ and S^{P^3} is obtained as given in Table 7, where $P^3 = [\{(\mathtt{s}, 1), (\mathtt{d}, 2), (\mathtt{n}, 2), (\mathtt{t}, 3), (\mathtt{l}, 1)\}, \{(\mathtt{l}, 2), (\mathtt{h}, 1), (\mathtt{n}, 1)\}, \{(\mathtt{y}, 1), (\mathtt{n}, 1)\}]$. Since S^{P^3} is unpartitionable, the whole process terminates and $P = P^3$ is a partition reduct. A more comprehensive version of S^{P^3} is given in Table 8.

Table 3
S^{P^1}

U	O^{P^1}	T^{P^1}	C^{P^1}	d
x_1	1	Low	1	Suspicious
x_2	Doctor	1	1	Yes
x_3	Nurse	1	1	Yes
x_5	Teacher	1	1	Suspicious
x_7	1	1	1	no

Table 4
$S_B^{P^1}$

U	$(O^{P^1},1)$	$(O^{P^1},\mathtt{d})$	$(O^{P^1},\mathtt{n})$	$(O^{P^1},\mathtt{t})$	$(T^{P^1},1)$	$(T^{P^1},\mathtt{l})$	$(C^{P^1},1)$	d
u_1	1	0	0	0	0	1	1	Suspicious
u_2	0	1	0	0	1	0	1	Yes
u_3	0	0	1	0	1	0	1	Yes
u_5	0	0	0	1	1	0	1	Suspicious
u_7	1	0	0	0	1	0	1	No

Table 5
S^{P^2}

U	O^{P^2}	T^{P^2}	C^{P^2}	d
u_1	1	2	1	Suspicious
u_2	2	1	1	Yes
u_5	Teacher	1	1	Suspicious
u_7	1	1	1	No

Table 6
$S_B^{P^2}$

U	$(O^{P^2},1)$	$(O^{P^2},2)$	$(O^{P^2},\mathtt{t})$	$(T^{P^2},1)$	$(T^{P^2},2)$	$(C^{P^2},1)$	d
u_1	1	0	0	0	1	1	Suspicious
u_2	0	1	0	1	0	1	Yes
u_5	0	0	1	1	0	1	Suspicious
u_7	1	0	0	1	0	1	No

Table 7
$S^P = S^{P^3}$

U	O^{P^3}	T^{P^3}	C^{P^3}	d
u_1	1	2	1	Suspicious
u_2	2	1	1	Yes
u_5	3	1	1	Suspicious
u_7	1	1	1	No

Table 8
A more comprehensive version of S^{P^3}

U	O^{P^3}	T^{P^3}	C^{P^3}	d
u_1	{Student, lawyer}	{Low}	{Yes, no}	Suspicious
u_2	{Doctor, nurse}	{Normal, high}	{Yes, no}	Yes
u_5	{Teacher}	{normal, High}	{Yes, no}	Suspicious
u_7	{Student, lawyer}	{Normal, high}	{Yes, no}	No

3.2. The optimal single-group partition problem

The example presented above indicates that the whole process is essentially recursive. Hence we now focus on the first round of the process, i.e., the computation of P^1.

As shown in Table 3, any conditional attribute has exactly one new value, that corresponds to one or more initial values. For example, for occupation, 1 corresponds to both `student` and `lawyer`, while for cough, 1 corresponds to both `yes` and `no`. That is, attribute values for each attribute form exactly one (single) new group. Hence, we introduce a special form of the partition scheme.

Definition 6. A partition scheme $P = [P_1, \dots, P_{|C|}]$ of S is called a single-group partition scheme (SGPS) if for any $i \in \{1, \dots, |C|\}$, $|W_{a_i}| = 1$.

Given an SGPS $P = [P_1, \dots, P_{|C|}]$, for any $i \in \{1, \dots, |C|\}$, P_i essentially divides V_{a_i} into two adisjoint subsets $V_{a_i}^F$ and $V_{a_i}^G$, and

$$P_i(v) = \begin{cases} v & \text{if } v \in V_{a_i}^F, \\ k & \text{if } v \in V_{a_i}^G, \end{cases} \tag{6}$$

where $k \in W_{a_i}$. According to Definition 6, $V_{a_i}^G \neq \emptyset$. Hence, any SGPS P can also be represented by a set of attribute value pairs, i.e., $P = \{(a_i, v) | i \in \{1, \ldots, |C|\}$ and $v \in V_{a_i}^F\}$.

For example, partition scheme P^1 is also an SGPS and it can be represented by $P^1 = R^1 = \{(O, \mathtt{d}), (O, \mathtt{n}), (O, \mathtt{t}), (T, \mathtt{l})\}$.

With this form of SGPS we can define single-group partition reducts as follows:

Definition 7. Any SGPS P is called a single-group partition reduct (SGPR) of S iff P is consistent and any $P' \subset P$ is not consistent.

The set of SGPR of S is denoted by $\mathrm{SGR}(S)$. A SGPR P is *optimal* iff $\mathrm{rank}(S^P)$ is minimal. We consider the following problem:

Problem 8. *Optimal single-group partition (OSGP)*
Input: A decision table $S = (U, C, \{d\})$ where all attributes are symbolic.
Output: An optimal SGPR P of S.

In fact, any SGPS $P \subseteq C_B$ is an attribute subset of S_B, and C^P is the set of conditional attributes of S^P. We have the following lemma:

Lemma 9. *For any SGPS $P \subseteq C_B$,*

$$\mathrm{Ind}(P) = \mathrm{Ind}(C^P). \tag{7}$$

Proof. For any $x_i, x_j \in U$,

$$\begin{aligned}
&(x_i, x_j) \in \mathrm{Ind}(P) \iff \forall (a, v) \in P, (a, v)(x_i) = (a, v)(x_j) \\
&\iff \forall a \in C, \{a\}^P(x_i) = \{a\}^P(x_j) \iff \forall a \in C, (x_i, x_j) \in \mathrm{Ind}(\{a\}^P) \\
&\iff (x_i, x_j) \in \bigcap_{a \in C} \mathrm{Ind}(\{a\}^P) = \mathrm{Ind}(C^P). \quad \square
\end{aligned}$$

According to Definition 7 and Lemma 9, we have the following theorem:

Theorem 10

$$\mathrm{SGR}(S) = \mathrm{Red}(S_B). \tag{8}$$

Therefore, the OSGP-problem of S (constructing an SGPR P) is converted into a reduction problem of S_B (selecting an attribute subset P from C_B).

Moreover, Eq. (6) indicates

$$\mathrm{rank}(S^P) = \sum_{1 \leqslant i \leqslant |C|} (V_{a_i}^F + 1) = |P| + |C|. \tag{9}$$

Finally, according to the definition of optimal metrics of reducts and SGPR, we have the following corollary:

Corollary 11. *The OSGP-problem of S is equivalent to the OR-problem of S_B.*

For example, since R^1 is an optimal reduct of S_B, according to Theorem 10, $P^1 = R^1$ is an optimal SGPS of S.

Also note that $R^1 \cap \{a_3\}_B = \emptyset$, hence $a_3^{P^1}(u) \equiv 1$, indicating $a_3^{P^1}$ is reduced.

Since the reduct problem is NP-complete, the following corollary holds:

Corollary 12. *The OSGP-problem is NP-complete.*

It is also easily seen that the single-group partition problem is more general than the reduct problem, yet more specific than the partition problem. The following theorem is straightforward:

Theorem 13. *If $V_{a_i} = 2$ for any $i \in \{1, \ldots, |C|\}$, the OSGP-problem coincides with the OR-problem.*

3.3. The algorithm

3.3.1. Algorithm structure

As indicated in Section 3.2, an (optimal) SGPR or the set of all SGPRs, of a decision table can be obtained by following three steps, namely, scaling, reduction and converting back. In most cases, however, an (optimal) SGPR is not a partition reduct.

The main structure of the algorithm therefore, involves repeating these three steps until the new decision table is unpartitionable. In other words, an SGPR of the new decision table is computed recursively with new values such as $2, 3, \ldots$ should be assigned to k in Eq. (6). In so doing, a partition reduct or even the set of all partition reducts can be obtained.

3.3.2. The usefulness of M-relative reduct

In the reduction step, the reduct for partition purposes should not be chosen randomly. In the example, R^1 is a reduct of S_B, and also an SGPR of S. $S^{R^1} = S^{P^1}$ and $\text{rank}(S^{R^1}) < \text{rank}(S)$. However, R^1 is also a reduct of $S_B^{P^1}$, and if used again as an SGPR of S^{P^1}, $(S^{P^1})^{R^1}$ would be equivalent to S^{P^1}. In fact, the only difference between S^{P^1} and $(S^{P^1})^{R^1}$ is in that all 1s in the former decision table are replaced by 2s in the latter. In the worst case, the whole process would enter a infinite loop if R^1 is always chosen as the SGPR of the new decision table.

Hence, the concept of *M*-relative reduct should be introduced to control the computation of SGPRs and ensure quick convergence of the algorithm. *M* is deliberately set such that new attribute values introduced can never be replaced by others. In the example, since 1 has replaced `student` as the occupation, it should not be replaced again by any other new values. Consequently, we let $M^2 = \{(O,1),(T,1)\}$ for the computation of R^2 and $M^3 = \{(O,1),(T,1),(O,2)\}$ for the computation of R^3.

Another important aspect of this approach is that once all attribute value pairs have been processed, i.e., respective values replaced by new ones, the algorithm should terminate. A two-dimension vector *H* is used to record unprocessed attribute value pairs.

3.3.3. The computation of partition schemes

According to our discussion in Section 3.2, $R^1, R^2, R^3, \ldots$ could be taken as SGPRs of $S, S^{P^1}, S^{P^2}, \ldots$, respectively. That is, $S^{P^1} = S^{R^1}, S^{P^2} = (S^{R^1})^{R^2}, S^{P^3} = ((S^{R^1})^{R^2})^{R^3}, \ldots$

The computation of $P^1, P^2, P^3, \ldots$ is given in the algorithm.

3.3.4. Algorithm description

The algorithm is given in Algorithm 1. It should be noted that we can compute $S_B^{P^i}$ without computing S^{P^i}, but for completeness we still list the pseudocode.

Algorithm 1. *The reduction-based symbolic value partition algorithm*

ReductionBasedSymbolicValuePartition $(S = (U, C, \{d\}))$
{**input:** *A decision table S.*}
{**output:** *A partition reduct P.*}
//Initialize. M^i *is used for M-relative reduct.*
Step 1. $M^1 = \emptyset$;
//The initial partition scheme P^0. *In fact* $S^{P^0} = S$.
Step 2. $P^0 = [P_1^0, \ldots, P_{|C|}^0]$ *where* $P_i^0(v_i) = v_i$ *for any* $i \in \{1, \ldots, |C|\}$ *and* $v_i \in V_{a_i}$;
//Initialize unprocessed attribute-value pairs for each attribute.
//Now all attribute-value pairs are unprocessed.
Step 3. *for* $(i = 1; i \leqslant |C|; i++)$ $H_i^0 = \{a_i\}_B$;
//Attack the OSVP-problem by recursively attacking the OSGP-problem.
Step 4. *for* $(i = 1; ; i++)$ *begin*
*//**scaling.***
Step 4.1 *compute* $S_B^{P^{i-1}}$;
*//**Reduction.***
Step 4.2 R^i = *an optimal M-relative reduct of* $S_B^{P^{i-1}}$ *where* $M = M^i$;
Step 4.3 $M^{i+1} = M^i$;*//Initialize* M^{i+1}.
Step 4.4 *for* $(j = 1; j \leqslant |C|; j++)$ *begin*
//Compute P^i.
Step 4.4.1 $\forall (a_j, v) \notin H_j^{i-1} - R^i$, $P_j^i(v) = P_j^{i-1}(v)$;
Step 4.4.2 $\forall (a_j, v) \in H_j^{i-1} - R^i$, $P_j^i(v) = i$;
Step 4.4.3 $H_j^i = H_j^{i-1} \cap R^i$*//Remove processed attribute-value pairs*
//Compute M^{i+1}
Step 4.4.4 *if* $(H_j^i \neq \emptyset)$ $M^{i+1} = M^{i+1} \cup \{(a_j, i)\}$;
end*//of for j*;
*//**Converting back to a "normal" decision table.***
Step 4.5 *compute* S^{P^i} *where* $P^i = [P_1^i, \ldots, P_{|C|}^i]$;
//See if all attribute-value pairs have been processed
Step 4.6 if $H^i = \bigcup_{j=1}^{|C|} H_j^i = \emptyset$ *break*; *end;//of for i*
Step 5. $P = P^i$, *return P*;

3.4. Algorithm analysis

In this subsection, we analyze the set of all possible outputs of the algorithm, explain why the *optimal M-reduct* should be computed in Step 4.2, and address some implementation issues.

3.4.1. Algorithm output

Step 4.2 requires that "R^i = an optimal$M$-relative reduct of $S_B^{P^{i-1}}$, where $M = M^i$". Now we do away with this requirement by removing *optimal* from the assessment and investigate the output of the algorithm.

Given a partition reduct P, it is straightforward to trace back the algorithm to construct the respective P^i, M^i and R^i. Hence, we have the following property:

Property 14. *Any $P \in \mathrm{PR}(S)$ can be the output of the algorithm.*

On the other hand, the output of the algorithm has the following property:

Property 15. *If $R^i \in \mathrm{Red}(S_B^{P^{i-1}}, M^i)$ for any $1 \leqslant i \leqslant K$, the algorithm output satisfies $P \in \mathrm{PR}(S)$.*

Integrating these two properties we obtain

Corollary 16

$$\bigcup_{i \in \{1,\ldots,K\}, R^i \in \mathrm{Red}(S_B^{P^{i-1}}, M^i)} \{P^K\} = \mathrm{PR}(S). \tag{10}$$

Corollary 16 indicates that if we execute the algorithm sufficient times and choose different M-relative reducts during the process, we can generally obtain $\mathrm{PR}(S)$. In other words, the set of all possible outputs of the algorithm is just the set of all partition reducts.

3.4.2. Optimal substructure

Since the goal of the OSVP-problem is to construct $P = P^K$ such that $\mathrm{rank}(S^{P^K})$ is minimal, it is natural to choose the optimal solution locally, i.e., the solution of the OSGP-problem. Thus Step 4.2 required that the M-relative reduct be *optimal*. As such, the algorithm is a *greedy* algorithm, and we need to explain why locally optimal solution can result in globally sub-optimal solution. The following theorem provides a partial reason.

Theorem 17. *The OSVP-problem has an optimal substructure property.*

Proof. Let $P = [P_1, P_2, \ldots, P_{|C|}]$ be an optimal partition reduct of S, $P' = [P'_1, P'_2, \ldots, P'_{|C|}]$ where $P'_i = (P_i - \{(v,1)|(v,1) \in P_i\}) \cup \{(1,1)$ for $i \in \{1,\ldots,|C|\}$, we can see that $(S^{P^1})^{P'} = S^P$. We need to prove that P' is an optimal partition reduct of S^{P^1}. Suppose there is another partition reduct $P'' = [P''_1, P''_2, \ldots, P''_{|C|}]$ of S^{P^1} such that $\mathrm{rank}((S^{P^1})^{P''}) < \mathrm{rank}((S^{P^1})^{P'})$. We can then construct another partition scheme $P^x = [P^x_1, P^x_2, \ldots, P^x_{|C|}]$ where $P^x_i = (P^1_i - \{(v,v)|(v,v) \in P^1_i\}) \cup (P''_i - \{(1,1)\})$, and $S^{P^x} = (S^{P^1})^{P''}$. This in turn gives

$$\mathrm{rank}(S^{P^x}) = \mathrm{rank}((S^{P^1})^{P''}) < \mathrm{rank}((S^{P^1})^{P'}) = \mathrm{rank}(S^P), \tag{11}$$

which means that P is not an optimal partition reduct and contradicts the assumption. Hence, an optimal solution P of the OSVP-problem of S contains the optimal solution P' of the same problem of S^{P^1}, and the proof is complete. □

In Step 4.1 an optimal M-relative reduct instead of an optimal reduct is computed for two reasons. The first reason was given in Section 3.3.2, while the second reason is that the optimal M-relative reducts computed in the algorithm are also optimal reducts. This is given by the following property:

Property 18. *R^i is an optimal reduct of $S_B^{P^{i-1}}$.*

Unfortunately, the OSVP-problem does not have the greedy-choice property. In what follows, we discuss a property that indicates that the algorithm can compute at least suboptimal partition reducts.

In the algorithm, H_j^{i-1} where $1 \leqslant i \leqslant K$ is the set of all unprocessed attribute value pairs before the ith round of Step 4, and $(H_j^{i-1} - H_j^i)$ is the set of all attribute value pairs processed in the ith round of Step 4. Since R^i is optimal, the following property holds.

Property 19. *Given $1 \leqslant i < k \leqslant K$, for any $1 \leqslant j \leqslant |C|$,*

$$|H_j^{i-1} - H_j^i| \geqslant |H_j^{k-1} - H_j^k|. \tag{12}$$

In the first round of the example, two Occupation values, namely, `student` and `lawyer` are partitioned into one group, we have $|H_1^0 - H_1^1| = 2$. In the next two rounds, we have $|H_1^1 - H_1^2| = 2 \leqslant 2$ and $|H_1^2 - H_1^3| = 1 \leqslant 2$.

This property also indicates that if Eq. (12) does not hold, then R^k is not an optimal M-relative reduct of $S^{P^{j-1}}$, and we should trace back to the kth round for a better solution to R^k.

3.4.3. Complexity analysis

In most applications, K is small. Table 10 gives some representative experimental results. Moreover, $S_B^{P^{i-1}}$ has far fewer attributes and objects (duplicate objects having been removed) than $S_B^{P^i}$. Hence, both the time and space complexities of the algorithm are determined by the first round, i.e., the reduct computation of S_B.

Any reduction algorithm can be employed. If we use the entropy method [11], the time complexity is

$$O(|U|^2 \text{rank}(S) + |U|^3). \tag{13}$$

A very interesting approach would be to borrow the idea of MD-heuristic for the reduct computation. This is feasible because S_B contains only binary conditional attributes, and thus the optimal discretization (OD) problem is equivalent to an OR-problem. If we use this approach, according to the complexity given in [15], the space complexity of our algorithm is

$$O(|U| \text{rank}(S)), \tag{14}$$

while the time complexity is

$$O(\text{rank}(S)|U|(|R| + \log|U|)), \tag{15}$$

where R is the reduct of S_B. In most applications $|R| \ll |U|$ and $\text{rank}(S) \ll |U||C|$, hence this approach is applicable.

4. Experiments with data

In this section, we investigate the performance of the algorithm with respect to: (1) generalization ability, which is evaluated by $\text{rank}(S^P)$ and (2) predication ability, which is evaluated by both the coverage and the F-measure of respective rule sets.

Two other approaches are compared with the partition approach: (1) the basic approach, in which only continuous attribute discretization is employed and (2) the attribute reduction approach.

4.1. The experimental setup

4.1.1. Datasets information

The eight datasets obtained from the UC Irvine ML repository [26] to test our algorithm are

(1) Monks dataset, including three training datasets and respective testing datasets (Monk1, Monk2 and Monk3).
(2) Mushroom database.
(3) Nursery database.
(4) Letter image recognition dataset (LetterRec), the first 16,000 items of which were used for training.
(5) Statlog project heart disease dataset (heart).
(6) Iris plants dataset (iris).

Basic information of these datasets is given in Table 9, where "Cont." denotes the number of continuous attributes in the dataset.

4.1.2. Experiment process

As some datasets contain continuous attributes, a discretization process was used to convert these attributes into discrete ones. MD-heuristic implemented in RSES 2.2 was employed for this purpose. For simplicity, the decision table after discretization, instead of the initial decision table, is denoted by S.

The exhaustive algorithm implemented in RSES 2.2 was employed to obtain all reducts. Since different optimal reducts may produce new decision tables with different ranks, the optimal reduct R with least $\text{rank}(S^R)$, where $S^R = (U, R, \{d\})$, was chosen.

The RBSVP algorithm was employed on S to produce S^P.

To keep the influence of the rule generation algorithm to a minimal, the exhaustive algorithm was employed to produce decision rules for all datasets except mushroom, where LEM2 was employed with the cover parameter set to 1. Conflicts were resolved by standard voting.

For datasets with specified training data, the number of decision rules were computed based on the training data, while for other datasets the computation was based on the entire dataset.

Table 9
Basic information of datasets

Dataset	$\lvert C\rvert$	$\lvert U\rvert$	$\lvert V_d\rvert$	Cont.
Monks1 (train)	6	124	2	0
Monks2 (train)	6	169	2	0
Monks3 (train)	6	122	2	0
Mushroom	22	8416	2	0
Nursery	8	12,960	3	0
LetterRec (train)	16	16,000	26	0
Heart	13	270	2	6
Iris	4	150	3	4

Furthermore, for datasets without specified training sets, CV-10 was employed to obtain respective rule coverage and F-measure (F).

The evaluation metrics include rule coverage (cov.) and F-measure (F) since high predication accuracy cannot prove the usefulness of the rule set while the coverage is low.

4.2. Analysis of the results

4.2.1. Analysis of ranks

Experimental results for the comparison of different generalization methods are given in Table 10.

It is easily seen that

$$\mathrm{rank}(S) \geqslant \mathrm{rank}(S^R) \geqslant \mathrm{rank}(S^P).$$

The cardinalities of conditional attribute sets has also been given since in some cases $\mathrm{rank}(S^P) - |C|$ instead of $\mathrm{rank}(S^P)$ may give a better reflection of the partition quality. For example, in the mushroom dataset, S^P has only seven "valid" conditional attributes, all of which are binary.

For all datasets K is small, indicating that only a few rounds are needed to obtain the partition reduct. Note that in LetterRec the cardinality of all attributes is 16; consequently, only 5 rounds (of which the last round essentially does nothing) are needed.

4.2.2. Analysis of the classification results using the generalized data sets

The respective results of rules generated in these datasets are given in Table 11.

For the Monks datasets, the RBSVP algorithm assists in obtaining smaller rule sets than the attribute reduction approach, which in turn offers a significant improvement over the direct approach. In addition, in 2 out of 3 Monks datasets the RBSVP algorithm performed the best in terms of the F-measure. Even for the last Monks dataset, the F-measure of the proposed approach is only slightly worse than the direct approach.

For the mushroom dataset, more rules were obtained with the introduction of attribute reduction. The RBSVP algorithm does not, however, incur any such problem here. In fact, the rule learning process is much faster after symbolic value partition.

For three datasets, namely nursery, heart and iris, attributes in S cannot be reduced further, hence the results corresponding to S and S^R are the same. However, the RBSVP algorithm is still effective here in that it helps produce better results.

For the LetterRec dataset, the results worsened with the introduction of attribute reduction, and even worse results were obtained with the RBSVP algorithm. There are at least two reasons for this. First, attributes in this dataset are essentially *ordered* rather than *symbolic*; and second, different attributes represent grey values of different points, that are not dependent.

Table 10
Comparison of the ranks of eight decision tables and their respective S^R and S^P

Dataset	Rank(S)	Rank(S^R)	Rank(S^P)	K
Monk1 (train)	17	13	11	3
Monk2 (train)	17	17	12	2
Monk3 (train)	17	15	15	4
Mushroom	116	35	29	2
Nursery	27	27	25	5
LetterRec (train)	256	160	56	5
Heart	25	25	24	3
Iris	11	11	10	2

Table 11
Comparison of rules (bold values indicate the best results)

Dataset	S			S^R			S^P		
	Rules	Cov.	F	Rules	Cov.	F	Rules	Cov.	F
Monk1	161	**1**	0.866	48	0.743	0.991	**10**	**1**	**1**
Monk2	247	**1**	0.736	95	0.773	0.871	**36**	0.995	**0.947**
Monk3	135	**1**	**0.944**	46	0.752	0.908	**39**	0.903	0.918
Mushroom	**14**	**1**	**1**	24	**1**	**1**	**14**	**1**	**1**
Nursery	638	**1**	0.983	638	**1**	0.983	**458**	**1**	**0.995**
LetterRec	3725	**0.661**	**0.725**	**3336**	0.615	0.692	5632	0.622	0.635
Heart	918	**1**	0.796	918	**1**	0.796	**807**	**1**	**0.826**
Iris	18	**1**	0.967	18	**1**	0.967	**15**	**1**	**0.973**

5. An approach of Nguyen

H.S. Nguyen and S.H. Nguyen [15,21] proposed the following an interesting partition approach. Suppose that $x_1, x_2 \in U$, $a \in C$, $a(x_1) = v_1$ and $a(x_2) = v_2$. If $v_1 \neq v_2$, then x_1 and x_2 are discerned by a, or more precisely, by the triple $a_{v_1}^{v_2} = (a, v_1, v_2)$. Hence, we can construct a new type of discernibility matrix, elements of which are triple sets instead of attribute sets.

An example decision table and the corresponding new type discernibility matrix are given in Table 12.

By using the discernibility function [25], two prime implicants can then be found:

$$R_1 = a_{\mathtt{w}}^{\mathtt{b}} \wedge a_{\mathtt{b}}^{\mathtt{g}} \wedge b_{\mathtt{w}}^{\mathtt{i}} \wedge b_{\mathtt{w}}^{\mathtt{g}} \wedge b_{\mathtt{i}}^{\mathtt{p}} \wedge b_{\mathtt{p}}^{\mathtt{g}} \quad \text{and}$$
$$R_2 = a_{\mathtt{w}}^{\mathtt{b}} \wedge a_{\mathtt{w}}^{\mathtt{g}} \wedge a_{\mathtt{b}}^{\mathtt{g}} \wedge b_{\mathtt{w}}^{\mathtt{g}} \wedge b_{\mathtt{i}}^{\mathtt{p}} \wedge b_{\mathtt{p}}^{\mathtt{g}},$$

both being the shortest.Suppose that R_1 is chosen, we obtain the graph as depicted in Fig. 1.

Now the partition problem has been converted into a graph coloring problem, that is, finding coloring schemes using the least colors such that adjacent nodes have different colors. For the graph depicted in Fig. 1, two colors are sufficient, indicating a partition scheme $P_1 = [P_a, P_b]$, where

$$P_a(\mathtt{white}) = P_a(\mathtt{green}) = 1, \quad P_a(\mathtt{black}) = 2;$$
$$P_b(\mathtt{wood}) = P_b(\mathtt{plastic}) = 1, \quad P_b(\mathtt{iron}) = P_b(\mathtt{glass}) = 2;$$

and P_1 is a partition reduct of the decision table.

Unfortunately, this approach has the following drawbacks. First, the space complexity is very high, as indicated in [21]: "[t]he constructed Boolean formula has $O(knl^2)$ variables and $O(n^2)$ clauses, where l is the maximal value of $|V_a|$ for $a \in C$"; and second, the partition scheme obtained may not be a partition reduct. Suppose that R_2 is employed to generate the graph as depicted in Fig. 2; we then obtain a partition scheme $P_2 = [P_a, P_b]$, where

$$P_a(\mathtt{white}) = 1, \quad P_a(\mathtt{black}) = 2, \quad P_a(\mathtt{green}) = 3;$$
$$P_b(\mathtt{wood}) = P_b(\mathtt{plastic}) = 1, \quad P_b(\mathtt{iron}) = P_b(\mathtt{glass}) = 2;$$

which is not a partition reduct. In fact, it can be proven that any partition scheme P produced by this approach is *consistent*, but S^P is not necessarily *unpartitionable*.

Table 12
An example for Nguyen's approach

U	a	b	d
x_1	white	wood	0
x_2	black	iron	0
x_3	white	plastic	0 →
x_4	green	iron	1
x_5	black	plastic	1
x_6	white	glass	1

$M(S)$	x_1	x_2	x_3
x_4	$a_{\mathtt{w}}^{\mathtt{g}}, b_{\mathtt{w}}^{\mathtt{i}}$	$a_{\mathtt{b}}^{\mathtt{g}}$	$a_{\mathtt{w}}^{\mathtt{g}}, b_{\mathtt{i}}^{\mathtt{p}}$
x_5	$a_{\mathtt{w}}^{\mathtt{b}}, b_{\mathtt{w}}^{\mathtt{p}}$	$b_{\mathtt{i}}^{\mathtt{p}}$	$a_{\mathtt{w}}^{\mathtt{b}}$
x_6	$b_{\mathtt{w}}^{\mathtt{g}}$	$a_{\mathtt{w}}^{\mathtt{b}}, b_{\mathtt{i}}^{\mathtt{g}}$	$b_{\mathtt{p}}^{\mathtt{g}}$

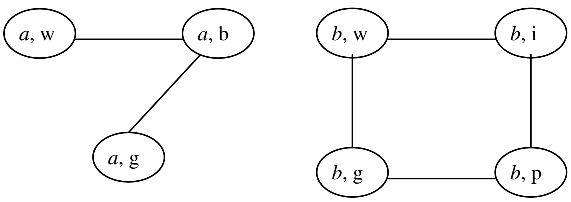

Fig. 1. The graph for coloring when R_1 is chosen.

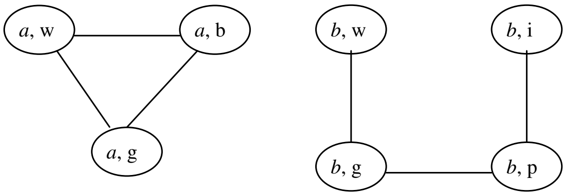

Fig. 2. The graph for coloring when R_2 is chosen.

6. Conclusions and further works

In this paper, we proposed the concept of partition reducts for the OSVP-problem. Since Nguyen's approach may not result in a partition reduct (see Section 5 for a counterexample), we developed an algorithm for this purpose. The new algorithm, called RBSVP, is both efficient (see Eqs. (14) and (15) for the space and time complexities) and tends to give suboptimal results (see Theorem 17, and Properties 15, 18 and 19). Experimental results also show that it assists in obtaining small rule sets with good performance for most datasets tested.

In further works we aim to extend this algorithm to suit mixed-mode data and/or decision tables with missing values, and apply it to applications such as intelligent information distribution. Generating rules in the partition process is also a very interesting issue.

References

[1] R. Jensen, Q. Shen, Semantics-preserving dimensionality reduction: rough and fuzzy-rough-based approaches, IEEE Transactions on Knowledge and Data Engineering 16 (12) (2004) 1457–1471.
[2] C.C. Chan, J.W. Grzymała-Busse, On the two local inductive algorithms: PRISM and LEM2, Foundations of Computing and Deision Sciences 19 (1994) 185–203.
[3] J.R. Quinlan, Induction of decision trees, Machine Learning 1 (1986) 81–106.
[4] R. Słowiński (Ed.), Intelligent Decision Support – Handbook of Applications and Advances of the Rough Sets Theory, Kluwer Academic Publishers, Dordrecht, 1992.
[5] Z. Pawlak, Rough sets, International Journal of Computer and Information Sciences 11 (1982) 341–356.
[6] J.G. Bazan, A. Skowron, P. Synak, Dynamic reducts as a tool for extracting laws from decision tables, in: Proceeding of the Symposium on Methodologies for Intelligent Systems, LNAI 869, 1994, pp. 346–355.
[7] M. Kryszkiewicz, Comparative studies of alternative type of knowledge reduction in inconsistent systems, International Journal of Intelligent Systems 16 (1) (2001) 105–120.
[8] Q. Liu, L. Chen, J. Zhang, F. Min, Knowledge reduction in inconsistent decision tables, in: X. Li, O.R. Zaïane, Z. Li (Eds.), ADMA, LNCS 4093, Springer, 2006, pp. 626–635.
[9] C. Xu, F. Min, Weighted reduction for decision tables, in: L. Wang, L. Jiao, G. Shi, X. Li, J. Liu (Eds.), Proceedings of The Third International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2006), LNCS 4223, Springer, 2006, pp. 246–255.
[10] J. Wróblewski, Finding minimal reducts using genetic algorithms, in: P.P. Wang (Ed.), JCIS'95, Wrightsville Beach, North Carolina, 1995, pp. 186–189.
[11] G. Wang, H. Yu, D. Yang, Decision table reduction based on conditional information entropy, Chinese Journal of Computers 25 (7) (2002) 1–8.
[12] Y.Y. Yao, Y. Zhao, J. Wang, On reduct construction algorithms, in: G.Y. Wang, J.F. Peters, A. Skowron, Y.Y. Yao (Eds.), Rough Sets and Knowledge Technology, First International Conference, RSKT 2006, Chongquing, China, July 24–26, 2006, Proceedings, LNCS 4062, Springer, 2006, pp. 297–304.
[13] F. Min, Q. Liu, C. Fang, J. Zhang, Reduction based symbolic value partition, in: ICHIT 2006 Electronic Proceeding, Soft Computing and Rough Sets, 2006, pp. 1–10.
[14] J.Y. Ching, A.K.C. Wong, K.C.C. Chan, Class-dependent discretization for inductive learning from continuous and mixed-mode data, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (1995) 641–651.
[15] H.S. Nguyen, Discretization of Real Value Attributes, Boolean Reasoning Approach, Ph.D. Thesis, Warsaw University, Warsaw, Poland, 1997.
[16] L.A. Kurgan, K.J. Cios, CAIM discretization algorithm, IEEE Transactions on Knowledge and Data Engineering 16 (2) (2004) 145–153.
[17] J.R. Quinlan (Ed.), C4.5 Programs for Machine Learning, Morgan Kaufman Publisher, San Mateo, California, 1993.
[18] N.C. Berkman, Value Grouping for Binary Decision Trees, 1995.
[19] K.M. Ho, Reducing decision tree fragmentation through attribute value grouping: a comparative study, Journal Intelligent Data Analysis 4 (3–4) (2000) 255–274.
[20] Z. Pawlak, Some issues on rough sets, in: J.F. Peters, A. Skowron, J.W. Grzymała-Busse, B. Kostek, R.W. Świniarski, M.S. Szczuka (Eds.), Transactions on Rough Sets I, LNCS 3100, Springer-Verlag, Berlin, Heidelberg, 2004, pp. 1–58.
[21] S.H. Nguyen, Regularity Analysis and its Applications in Data Mining, Ph.D. Thesis, Warsaw University, Warsaw, Poland, 1999.
[22] Y.Y. Yao, A partition model of granular computing, Transactions on Rough Sets I (2004) 239–259.
[23] F. Min, Q. Liu, H. Tan, L. Chen, The *M*-relative reduct problem, in: G.Y. Wang, J.F. Peters, A. Skowron, Y.Y. Yao (Eds.), Rough Sets and Knowledge Technology, First International Conference, RSKT 2006, Chongquing, China, July 24–26, 2006, Proceedings, LNCS 4062, Springer, 2006, pp. 170–175.
[24] B. Ganter, R. Wille (Eds.), Formal Concept Analysis: Mathematical Foundations, Springer-Verlag, New York, 1999.
[25] A. Skowron, C. Rauszer, The discernibility matrices and functions in information systems, in: R. Słowiński (Ed.), Intelligent Decision Support – Handbook of Applications and Advances of the Rough Sets Theory, Kluwer Academic Publishers, Dordrecht, 1992, pp. 331–362.
[26] C.L. Blake, C.J. Merz, UCI Repository of Machine Learning Databases, 1998, <http://www.ics.uci.edu/~mlearn/mlrepository.html>.